CLAIMS

1. A method for providing a Web service by a plurality of Web domains hosted by a computer, through a single IP address, comprising:

a) For each of said domains, allocating a server having a unique domain name and said IP address, for providing said service;

b) Providing a wrapper, being a software module for intermediating between a client of said service and said servers via the standard communication protocol for communicating with each of said severs;

c) Upon receiving a request for connecting said client to the one of said servers in order to provide said service:

(i) Identifying the target domain name of said request by interacting between said client and said wrapper via said standard protocol;

(ii) Interacting between said wrapper and the server providing said service which is associated with said target domain name by said standard protocol;

(iii) Establishing a communication channel between said server and said client utilizing said standard protocol; and

(iv) Allowing said server to provide said service to said client.

2. A method according to claim 1, wherein the username phrase being used includes the username and the domain, and the domain name is separated from the username by one or more characters which do not conform with the standard characters allowed in a username in the standard protocol.

3. A method according to claim 2, wherein the username phrase is "user%domain" or "domain%user", in which "user" is the username, "domain" is the domain name, and "%" is any character which does not conform with the standard protocol for such phrasing purposes.

4. A method according to claim 1, wherein said Web services are chosen from among HTTP, FTP, POP3, SMTP, MIRC, Telnet, SSH, Rtelnet, and Shell.

5. A method according to claim 1, wherein each of said Web domains refer to a different Virtual Dedicated Server.

6. A method according to claim 1, wherein said computer system is a Unix-based system, any dialect of Unix, Solaris, Linux (Red Hat, Debian, SuSE, FreeBSD, etc.), AIX, HP/UX, Tru64, or Irix.

7. A method according to claim 1, wherein each domain has its own instance of the server.

8. A method according to claim 7, wherein the server(s) of some or all the domains share the same disk space.

9. A method according to claim 8, wherein only one instance of some or all of the server(s) resides at the Host, and being referenced by hard links from the domains.

10. A method according to claim 1, wherein the wrapper is kept active for the entire session.

11. A method according to claim 1, wherein the wrapper is kept active only until the requested server is identified, and the communication is handled to this server.

12. A method according to claim 1, further comprising providing a new shared library including additional functionality to the original shared library to which the standard communication protocol refers.

13. A method according to claim 12, wherein the additional functionality of the new shared library is added to the original shared library by hooking.

14. A method according to claim 12, further comprising providing a buffer to each socket, for retaining temporarily the information received from the client, and reading the data from said buffer if it is not empty, or from the socket if it is empty.

15. A method according to claim 14, further comprising ignoring any write command until the buffer is empty.

16. A method according to claim 1, wherein one encryption key is used for all domains on each Host.

17. A method according to claim 1, wherein the wrapper is provided with information related to secured services of the target domain in plain text.

18. A system for providing a Web service to a client by a plurality of Web domains hosted by a computer, through a single IP address, comprising:

- A server for providing said service for each of said domains; and
- A wrapper, for intermediating between said client and said servers, such that communicating with said client is carried out via the standard communication protocol.

where for each request for connecting said client said server said wrapper identifies the target domain name by interacting with said client via said standard protocol, interacts with the server associated with said target domain name via said standard protocol, and enables said server to provide said service to said client.

19. A system according to claim 18, wherein the wrapper is active for the entire session.

20. A system according to claim 19, wherein the wrapper is kept active only until the requested server is identified, and the communication is handled to this server.

21. A system according to claim 18, further comprising a new shared library including additional functionality to the original shared library to which the standard communication protocol refers.

22. A system according to claim 21, wherein the additional functionality of the new shared library is added to the original shared library by hooking.

23. A system according to claim 18, wherein the additional functionality includes retaining temporarily the information received from the client via a socket into a buffer, and reading the data from said buffer if it is not empty, or from the socket if it is empty.

24. A system according to claim 23, further comprising ignoring any write command until the buffer is empty.

25. A system according to claim 18, wherein one encryption key is used for all domains on each server.

26. A system according to claim 25, wherein the wrapper is provided with information related to secured services of the target domain in plain text.

27. A system according to claim 18, wherein each domain has its own instance of the server.

28. A system according to claim 27, wherein the server(s) of some or all the domains share the same disk space.

29. A system according to claim 28, wherein only one instance of some or all of the server(s) resides at the Host, and being referenced by hard links, from the domains.

30. A wrapper for handling the connection of clients to a plurality of Web domains hosted by a single Host in which said connection is handled over the standard communication protocol by providing a buffer to each socket for retaining temporarily the information received from the client.

31. A wrapper according to claim 30, further comprising providing the servers with additional functionality by hooking a new shared library to the original shared library.

32. A wrapper according to claim 31, wherein during the connection "read" commands read the data from the buffer if it is not empty, or the data from the socket, if said buffer is empty.

33. A wrapper according to claim 30, wherein the connection further comprising ignoring any write command until the buffer is empty.